



**SENAI** *Serviço Nacional  
de Aprendizagem  
Industrial*

# SQL

Structured Query Language (**SQL** – Linguagem de Consulta Estruturada) é uma linguagem da informática destinada a armazenar, manipular e obter dados armazenados em bases de dados relacionais;

- Surgiu em 1974 dentro da empresa IBM;



## SGBD escolhido: MySQL



OBS: Para instalar o MySQL vamos usar a ferramenta XAMP (Já trás em conjunto o MySQL (SGBD) e Apache (Servidor Web)).



# Tipo de dados

- Integer – Números inteiros
- Char(tamanho) – Caracteres com ocupação total
- Varchar(tamanho) – Caracteres com ocupação parcial
- Numeric(tamanho,decimais) – Define números reais com precisão decimal

OBS: Numeric = Decimal

- Date - data
- Time - hora
- Timestamp – Data e hora juntos

Os comandos são agrupados em 5 grupos:

## SQL

**DDL** Definição

**DCL** Controle

**DQL** Solicitações

**DML** Manipulação

**DTL** Transações

# Comandos DDL

- **CREATE** Criar banco, tabela, índice, visão
- **ALTER** Alterar estrutura
- **DROP** Excluir definitivamente
- **TRUNCATE** Limpar dados da tabela
- **RENAME** Renomear tabela/objeto

# Comandos DDL: Create

Cria tabelas, bancos de dados, índices, visões etc.

## Sintaxe básica:

```
CREATE TABLE nome_tabela (  
    coluna1 tipo_dado [NOT NULL],  
    coluna2 tipo_dado [NOT NULL],  
    ...  
    PRIMARY KEY (coluna)  
);
```

OBS: Inicialmente é necessário criar a base de dados `CREATE Database nomedabase;`

# Comandos DDL: Create

Cria tabelas, bancos de dados, índices, visões etc.

```
CREATE TABLE alunos (  
    id INT PRIMARY KEY AUTO_INCREMENT,  
    nome VARCHAR(100),  
    idade INT  
);
```

OBS: Inicialmente é necessário criar a base de dados `CREATE Database nomedabase;`

# Comandos DDL: Alter

Altera a estrutura de uma tabela existente.

Adicionar coluna

```
ALTER TABLE alunos ADD email VARCHAR(150);
```

Alterar tipo

```
ALTER TABLE alunos MODIFY idade SMALLINT;
```

Renomear tabela

```
ALTER TABLE alunos RENAME TO estudantes;
```

Renomear coluna

```
ALTER TABLE fabrica  
CHANGE COLUMN email e_mail varchar(100);
```

# Comandos DDL: Drop

Exclui tabelas, bancos de dados ou outros objetos.

Apaga tabela alunos

```
DROP TABLE alunos;
```

Apaga base de dados escola

```
DROP DATABASE escola;
```

# Comandos DDL: Truncate

Apaga **todos os dados** da tabela, mas **mantém a estrutura**.

Apaga dados de alunos

```
TRUNCATE TABLE alunos;
```

OBS: Muito mais rápido que DELETE, mas não pode ser desfeito.

# Comandos DDL: Rename

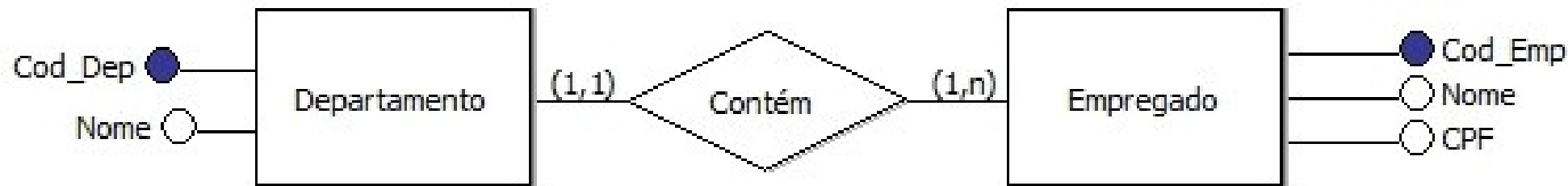
Renomeia objetos do banco (em alguns SGBDs é parte do ALTER).

Renomeia a tabela Alunos:

```
RENAME TABLE alunos TO estudantes;
```

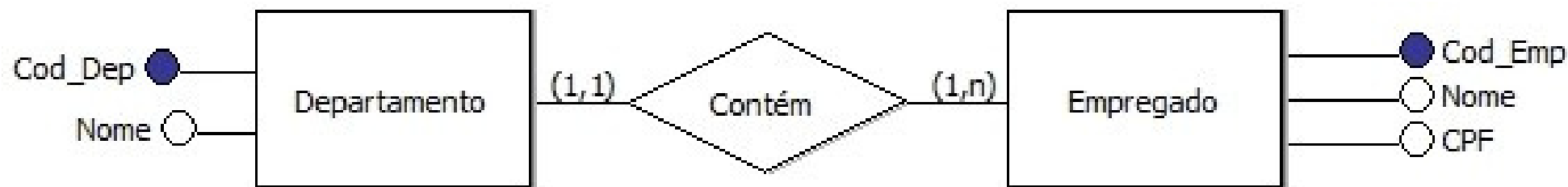
# Chave Estrangeira

- Permite a relação entre tabelas em um banco de dados.



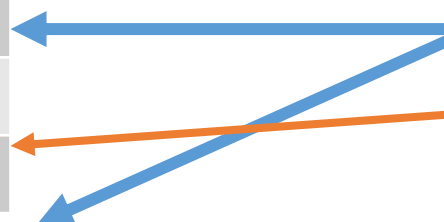
- Empregado vai receber a chave primária de Departamento.

# Chave Estrangeira



Cod_Emp	Nome	CPF	Cod_dep
10	Marcos	024.700.234-21	01
11	Mateus	5555	02
12	Maria	7777	01

Cod_dep	Nome
01	TI
02	RH



# Chave Estrangeira

```
CREATE TABLE nome_tabela (  
    coluna1 tipo_dado,  
    coluna2 tipo_dado,  
    ...  
    PRIMARY KEY (coluna),  
    FOREIGN KEY (coluna)  
        REFERENCES nome_tabela (coluna_referenciada),  
    UNIQUE (coluna)  
);
```

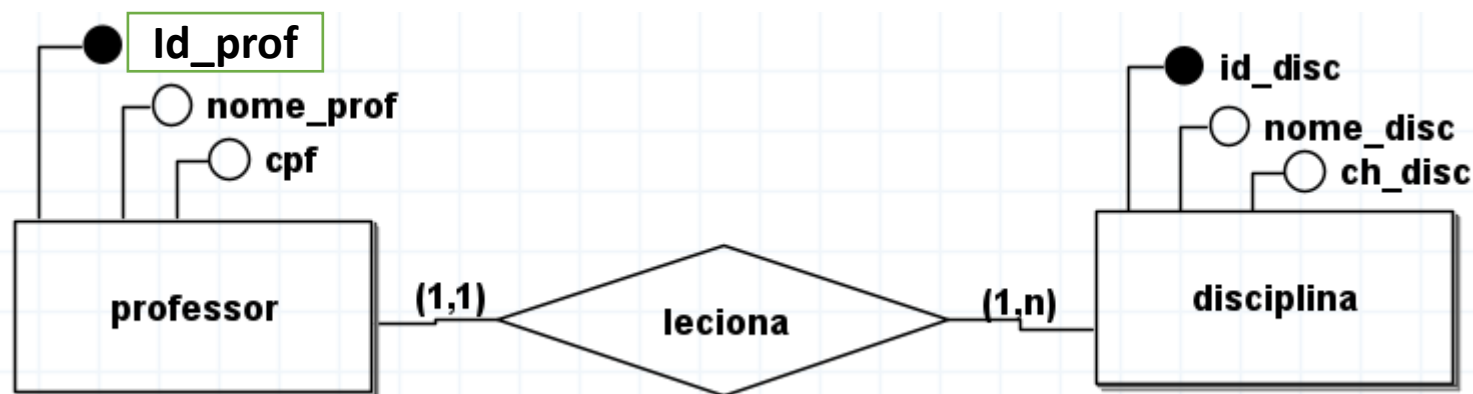
# Chave Estrangeira

OBS: A tabela Departamento já tem que estar no BD, para que a coluna possa ser referenciada

```
CREATE TABLE Empregado (  
  Cod_Emp INT NOT NULL auto_increment,  
  NOME VARCHAR(40),  
  CPF INT,  
  ID_Dep INT,  
  
  PRIMARY KEY (Cod_Emp),  
  FOREIGN KEY (ID_Dep) REFERENCES Departamento (Cod_Dep),  
  UNIQUE (CPF)  
);
```

# Exercício

- Crie as tabelas abaixo usando comandos SQL:



# Comandos DML (Manipulação)

- INSERT – Insere registros na tabela
- UPDATE - Altera registros
- DELETE - Exclui registros

# Comandos - **INSERT**

O comando **INSERT** adiciona novos dados em uma tabela.

**sintaxe:**

```
INSERT INTO nome_da_tabela (coluna1, coluna2, coluna3)  
VALUES (valor1, valor2, valor3);
```

Exemplo – Inserindo dados na tabela professor:

```
INSERT INTO professor (id_prof, nome_prof, cpf)  
VALUES (1, 'João Silva', '12345678900');
```

Id_prof	Nome_prof	cpf
1	João Silva	12345678900

# Comandos - **INSERT**

O comando **INSERT** adiciona novos dados em uma tabela.

Exemplo – Inserindo dados na tabela disciplina:

```
INSERT INTO disciplina (id_disc, nome_disc, ch_disc)  
VALUES (10, 'Banco de Dados', 80);
```

Id_prof	Nome_prof	cpf
10	Banco de Dados	80

# Comandos - UPDATE

**Modifica informações** de um ou mais registros existentes em uma tabela.

Exemplo – Atualizando o nome de um professor:

**Cuidado:** Se o comando `WHERE` for omitido, **todos os registros** da tabela serão alterados.

```
UPDATE professor  
SET nome_prof = 'João da Silva'  
WHERE id_professor = 1;
```

Id_prof	Nome_prof	cpf
1	João Silva	12345678900

Id_prof	Nome_prof	cpf
1	Marcos Silva	12345678900



# Comandos - DELETE

**Modifica informações** de um ou mais registros existentes em uma tabela.

Exemplo – Apagando professor:

**Cuidado:** Se o comando `WHERE` for omitido, **todos os registros** da tabela serão alterados.

```
DELETE FROM professor  
WHERE id_professor = 1;
```

Id_prof	Nome_prof	cpf
1	Marcos Silva	12345678900

Id_prof	Nome_prof	cpf



# Comandos - EXERCÍCIO

Crie as tabelas abaixo, e adicione pelo menos 5 registros para cada coluna.

